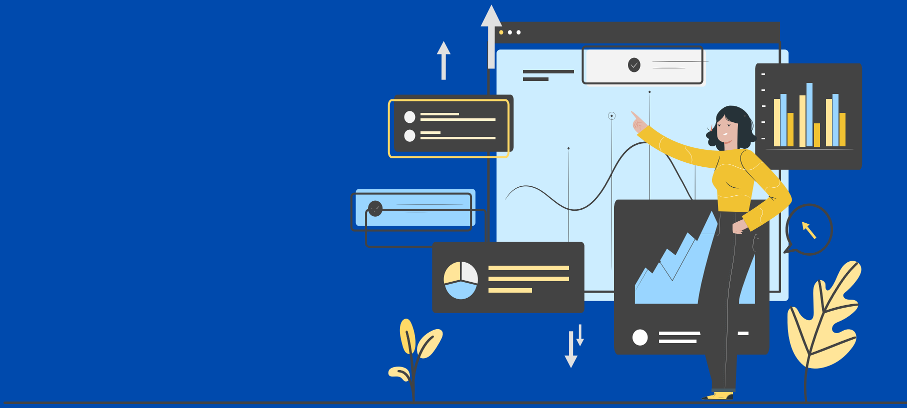
CSDS 313:

Title of Research: An examination on effect of Technical Analysis indicators on predicting stock movement.



Name: Kiet Nguyen, Zhuldyz Ualikhankyzy, Tu Pham
University: Case Western Reserve University
Date: December 5th, 2023

I. OVERVIEW

1

Dataset:

Historic data for the daily ten minute closing auction on the NASDAQ stock exchange market.

Application:
Predict the future price movements of stocks relative to the price future movement of a synthetic index.

2

Hypothesis:

Technical Analysis indicators outperform given features in predicting stock move.

Evaluating metrics:
2.1 Correlation
2.2 Mutual Information
2.3 Feature Importance of LightGBM

Feedback respond: define a plausible scope of work.

3

Methodology:

2.1 Data Cleaning using python pandas package
2.2 Data size reduction (time and number of stocks)
2.3 Training the model using LightGBM and n-fold (n=5) validation
2.4 Hypothesis testing

2. DATA LITERACY

	unique	cardinality	with_null	null_pct	1st_row	random_row	last_row	dtype
stock_id	False	200	False	0.00	0	89	199	int64
date_id	False	481	False	0.00	0	151	480	int64
seconds_in_bucket	False	55	False	0.00	0	450	540	int64
imbalance_size	False	2971863	True	0.00	3180602.69	2890625.99	1884285.71	float64
imbalance_buy_sell_flag	False	3	False	0.00	1	1	-1	int64
reference_price	False	28741	True	0.00	1.0	1.003	1.002	float64
matched_size	False	2948862	True	0.00	13380276.64	30679227.24	24073677.32	float64
far_price	False	95739	True	55.26	NaN	1.041	1.001	float64
near_price	False	84625	True	54.55	NaN	1.023	1.001	float64
bid_price	False	28313	True	0.00	1.0	1.002	1.002	float64
bid_size	False	2591773	False	0.00	60651.5	376.68	250081.44	float64
ask_price	False	28266	True	0.00	1.0	1.003	1.002	float64
ask_size	False	2623254	False	0.00	8493.03	3140.5	300167.56	float64
wap	False	31506	True	0.00	1.0	1.002	1.002	float64
target	False	15934	True	0.00	-3.03	-31.07	-6.53	float64
time_id	False	26455	False	0.00	0	8350	26454	int64
row_id	True	5237980	False	0.00	0_0_0	151_450_89	480_540_199	object

Overview on data:
Refer to Appendix 14 for data dictionary

~200 stocks, 481 trading dates, 55 time steps per series
=> 96,200 time series in training data
53,020 missing values
Expected due to missing stocks on some dates. → Missing full date data.

Variables:
time_id: permutation of seconds_in_bucket & date_id
row_id: concatenation of date_id, seconds_in_bucket, stock_id
target: target variable to predict
Other columns: feature variables

ORDER BOOK

Bid	Price	Ask
	10	1
2	9	
0	8	

Bid	Price	Ask
	10	1
	9	8
0	8	

$$\text{WAP} = \frac{BidPrice * AskSize + AskPrice * BidSize}{BidSize + AskSize}$$

Refer to Appendix slide 16, for further visualization on WAP's property and interaction.
WAP = weighted average price

AUCTION ORDER BOOK

Bid	Price	Ask
	10	1
3	9	2
4	8	4

Uncross price: 8
Matched size: 4 lots
Imbalance: 3 excess bids
→ 3 lot buy imbalance

far_price = 8
matched_size = 4 * ref price
imbalance_size = 3 * ref price
imbalance_buy_sell_flag = 1 (1 for buy-side imbalance, -1 for sell-side imbalance, 0 for no imbalance)

Definition:
Uncross price: Closing auction price
Far price: Hypothetical uncross price if auction ended now

COMBINED BOOK

Bid	Price	Ask
	10	2
5	9	2
4	8	4

The uncross price is 9
The matched size is 5
The imbalance would be 1 lot, in the sell direction.

2.1. DATA CLEANING

Removed "row_id" as it is completely unrelated to the target prediction.
The dataset has missing data, primarily due to certain stocks missing data on some days entirely. Therefore, we need to drop entire data of these time stamps for corresponding stocks.

STOCK_ID	69	73	78	79	99	102	135	150	153	156	199
# MISSING DAYS	37	1	4	181	1	295	191	59	70	37	88

2.2.FEATURE ENGINEERING

- Capture repetitive behavioral patterns that manifest in prices -> capture tradable opportunities.
- The signals indicators produce supplement standard feature data with market psychology and trading logic, enhancing opportunity for predictive modeling.

TECHNICAL ANALYSIS

I. RSI

2. MACD

3. BOLLINGER BANDS

The relative strength index (RSI) is a momentum indicator used in technical analysis. RSI measures the speed and magnitude of a security's prices changes to evaluate overvalued or undervalued conditions in the price of that security.

$$\mathrm{RSI} = 100 - \frac{100}{1 + \mathrm{RS}}$$

Moving average convergence/divergence is a trend-following momentum indicator that shows the relationship between two exponential moving averages (EMAs) of a security’s prices. The MACD line is calculated by subtracting the 26-period EMA from the 12-period EMA.

Bollinger Bands capture a dynamic volatility-based envelope around prices to judge extremes and turning points useful for forecasting. The width of bands adapts based on recent variance.

$$\mathrm{BOLU} = \mathrm{MA}(\mathrm{TP}, n) + m * \sigma[\mathrm{TP}, n]$$

$$\mathrm{BOLD} = \mathrm{MA}(\mathrm{TP}, n) - m * \sigma[\mathrm{TP}, n]$$

Refer to Appendix 16 for code snippet.

3. EXPLORATORY DATA ANALYSIS

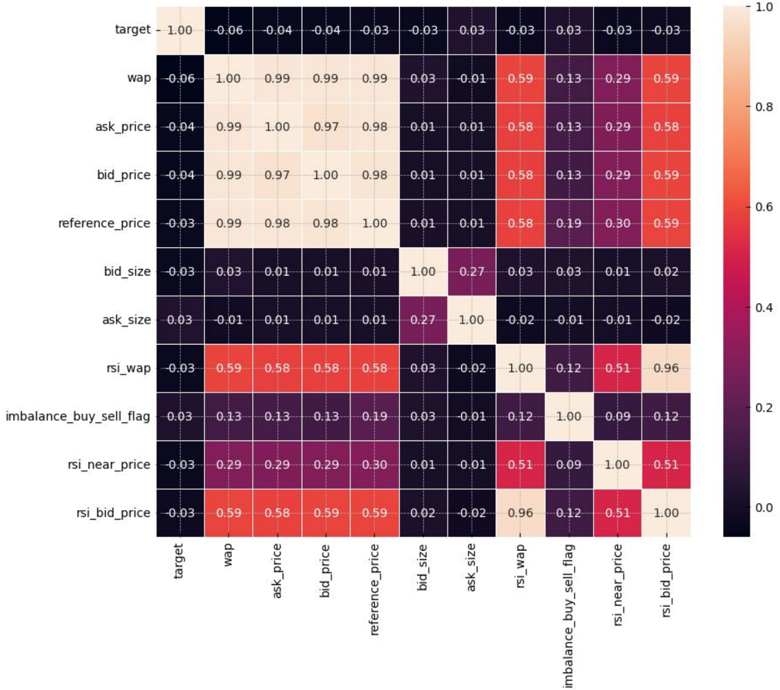


Figure 2: Heatmap capturing correlation among top 10 feature and the target

- New technical indicators such as RSI show noticeable linear correlation to raw price indicators like WAP, bid price etc., validating that they may capture valuable signal.
- However, among the top predictive features, there is low linear correlation observed with the target variable. This suggests that complex non-linear relationships underpin the mappings from inputs to target.
- Simple linear regression or correlation-based models may not be best suited, while nonlinear techniques like ensemble learning, neural networks etc. can better uncover intricate dependencies missed by linear analysis.

TOP 10 MUTUAL INFORMATION (MI)

Features	MI with target
stock_id	0.0465
time_id	0.0327
rsi_far_price	0.0258
bid_price	0.0237
matched_size	0.0236
rsi_near_price	0.0234
bid_size	0.0200
wap	0.0197
ask_price	0.0181
ask_size	0.0174

- The stock_id, time_id, rsi_far_price are likely to be more informative in predicting the target.
- The RSI in TA features are identified in top 10 features on MI with target. We will train model with & without the RSI to observe if it can improve model prediction.
- In this top 10, WAP was also the most (negatively) linearly correlated with target, so it has significant relationship with target in both linear and non-linear sense and should be taken into consideration.

4. FEATURE IMPORTANCE

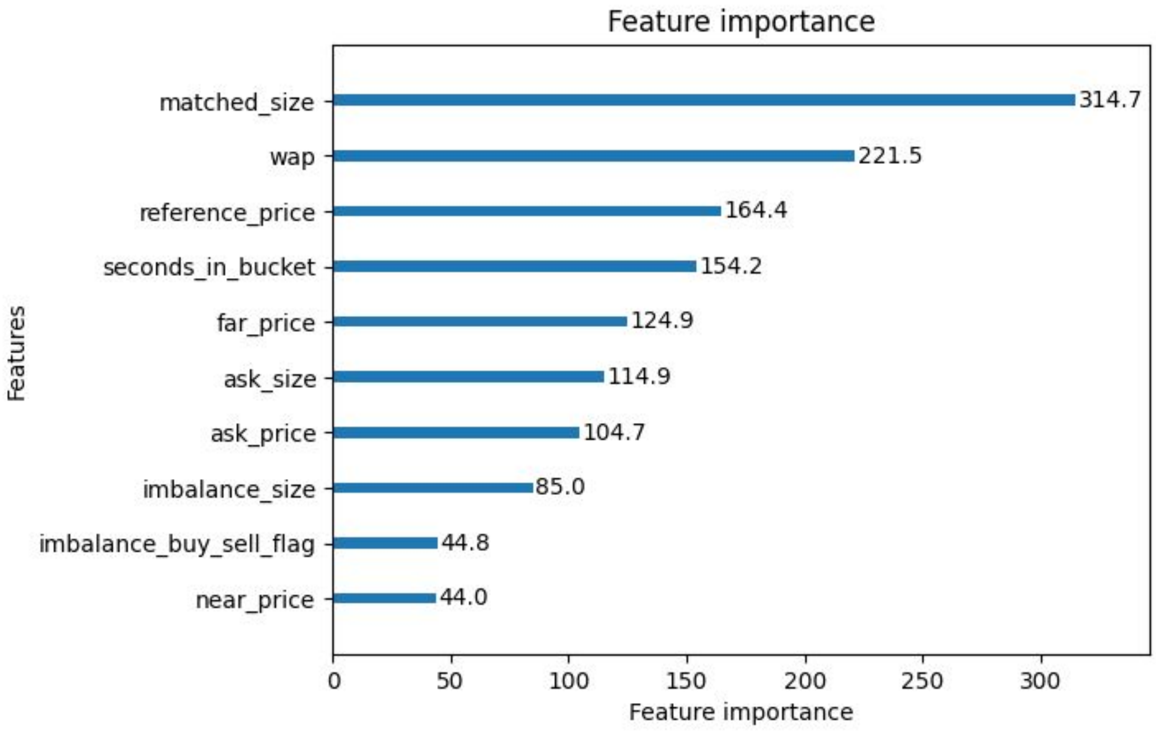


Figure 2: Feature importance of the model trained on 10 stock with the basic set of features

Feature importance in LightGBM is assessed using Gain (measuring improvement in model performance). Higher performance gain indicates greater importance for a feature.

4. RESULTS ANALYSIS

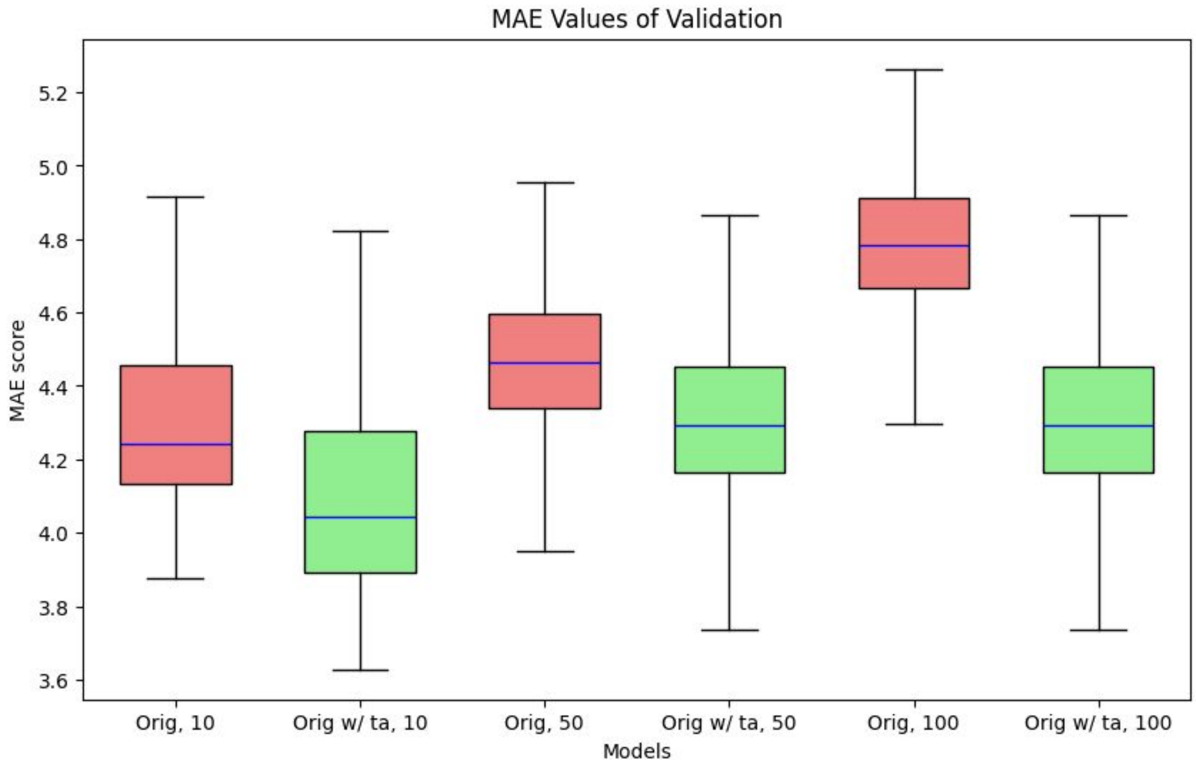


Figure 4: Mean Average Error of Validation for six different models

90%/10% - train to test splitting (due to extensive validation)
Trained the model using 10,50 and 100 stocks
Used one fiscal quarter, corresponding to 63 days
num_folds = 5
fold_size = 63 // num_folds
gap = 5
Learning_rate = 0.01

4. RESULTS ANALYSIS

	10 stocks	50 stocks	100 stocks
H0: MAE score of validation of the model with the technical analysis (TA) features is at least 10% lower than that of the model without the TA features H1: MAE score of validation of the model with the technical analysis (TA) features is less than 10% lower than that of the model without the TA features	✗	✗	✓
H0: MAE score of validation of the model with the technical analysis (TA) features is lower than that of the model without the TA features H1: MAE score of validation of the model with the technical analysis (TA) features is not lower than that of the model without the TA features	✓	✓	✓

According to the results of the Welch test, it is evident that, in general, all three cases exhibit statistically significantly lower results when Technical Analysis (TA) features are added to the feature set compared to the cases when only the original features were used. However, not all the cases of adding TA features hold when testing the improvement of the Mean Absolute Error (MAE) score by 10%.

5. CONCLUSIONS & FUTURE RESEARCH

Technical Analysis features significantly improve the the accuracy of stock-prediction model

Across all three experiments, the decrease of MAE scores in the models with TA features was statistically significant

The number of stocks affect the importance of features

Feature importance produced from the same set of features, but with different size of stocks differ from one another.

The gain from TA features increase with the increase of the data set size

Models with a larger data set produce considerably lower MAE results when TA features are introduced compared to models trained on smaller data sets

Evaluate whether the last two observations are caused by confounding effect

Use Lazy Predict to check whether there are better models than LightGBM

Carry out the training for each experiment iteratively to compensate for the randomness of splits

05. ACKNOWLEDGEMENTS

MEHMET KOYUTÜRK

Professor at Case Western Reserve University

REFERENCES:

Fernando, Jason. “Relative Strength Index (RSI) Indicator Explained with Formula.” Investopedia, Investopedia, www.investopedia.com/terms/r/rsi.asp. Accessed 20 Dec. 2023.

Hayes, Adam. “Bollinger Bands®: What They Are, and What They Tell Investors.” Investopedia, Investopedia, www.investopedia.com/terms/b/bollingerbands.asp#:~:text=A%20Bollinger%20Band%C2%AE%20is,be%20adjusted%20to%20user%20preferences. Accessed 20 Dec. 2023.

“Optiver - Trading at the Close.” Kaggle, www.kaggle.com/competitions/optiver-trading-at-the-close/data. Accessed 20 Dec. 2023.

Yang, Junwei. Explain the Data | Lightgbm Baseline | Kaggle, www.kaggle.com/code/a27182818/explain-the-data-lightgbm-baseline. Accessed 20 Dec. 2023.

Data dictionary

stock_id - A unique identifier for the stock. Not all stock IDs exist in every time bucket.

date_id - A unique identifier for the date. Date IDs are sequential & consistent across all stocks.

imbalance_size - The amount unmatched at the current reference price (in USD).

imbalance_buy_sell_flag - An indicator reflecting the direction of auction imbalance.
buy-side imbalance; 1
sell-side imbalance; -1
no imbalance; 0

reference_price - The price at which paired shares are maximized, the imbalance is minimized and the distance from the bid-ask midpoint is minimized, in that order.

matched_size - The amount that can be matched at the current reference price (in USD).

matched_size - The amount that can be matched at the current reference price (in USD).

far_price - The crossing price that will maximize the number of shares matched based on auction interest only. This calculation excludes continuous market orders.

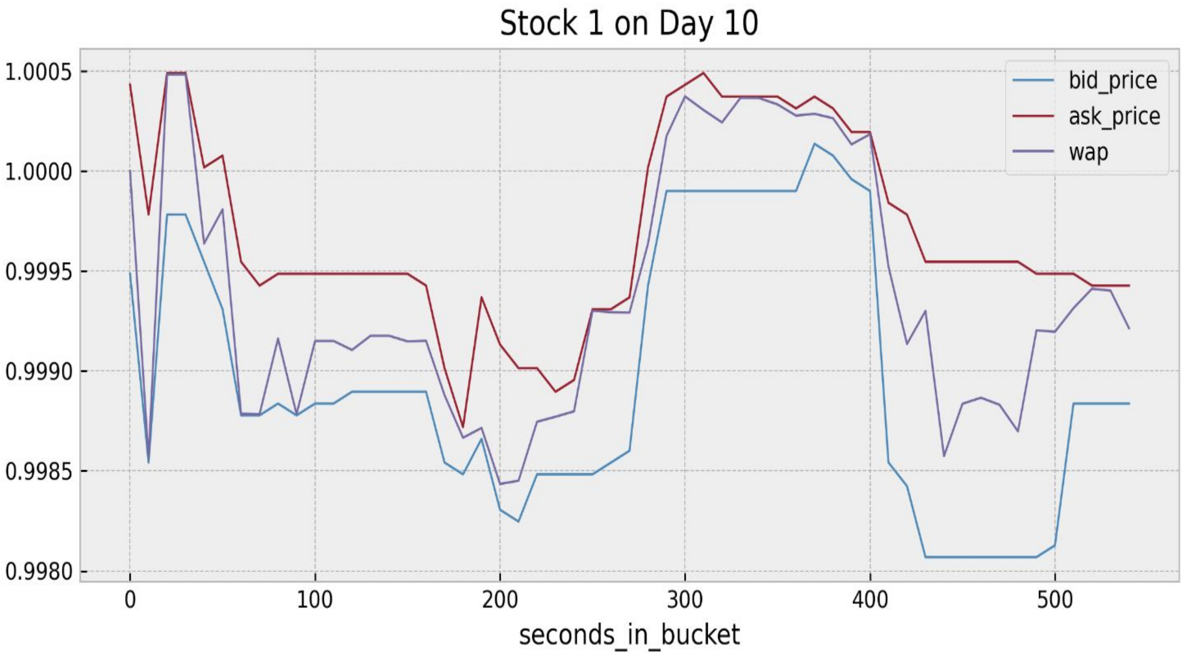
near_price - The crossing price that will maximize the number of shares matched based auction and continuous market orders.

[bid/ask]_price - Price of the most competitive buy/sell level in the non-auction book.

[bid/ask]_size - The dollar notional amount on the most competitive buy/sell level in the non-auction book.

target - The 60 second future move in the wap of the stock, less the 60 second future move of the synthetic index.

The unit of the target is basis points, which is a common unit of measurement in financial markets. A 1 basis point price move is equivalent to a 0.01% price move.



WAP's properties:

- wap falls between bid_price and ask_price
- Larger bid_size -> wap pushed towards ask_price
- Larger ask_size -> wap pushed towards bid_price
- But wap always stays within spread

In essence, wap represents a fair price estimate, positioned inside the bid-ask spread in proportion to relative sizes on buy and sell sides. It is tugged towards higher or lower equilibrium by imbalanced market activity, while remaining bounded by the marginal prices in place.

06. APPENDIX

RSI

```
def calculate_rsi(prices, period=14):
    rsi_values = np.zeros_like(prices)

    for col in prange(prices.shape[1]):
        price_data = prices[:, col]
        delta = np.zeros_like(price_data)
        delta[1:] = price_data[1:] - price_data[:-1]
        gain = np.where(delta > 0, delta, 0)
        loss = np.where(delta < 0, -delta, 0)

        avg_gain = np.mean(gain[:period])
        avg_loss = np.mean(loss[:period])

        if avg_loss != 0:
            rs = avg_gain / avg_loss
        else:
            rs = 1e-9  # or any other appropriate default value

        rsi_values[:period, col] = 100 - (100 / (1 + rs))

        for i in prange(period-1, len(price_data)-1):
            avg_gain = (avg_gain * (period - 1) + gain[i]) /
period      avg_loss = (avg_loss * (period - 1) + loss[i]) /
period      if avg_loss != 0:
                rs = avg_gain / avg_loss
            else:
                rs = 1e-9  # or any other appropriate default
value       rsi_values[i+1, col] = 100 - (100 / (1 + rs))

    return rsi_values
```

MACD

```
def calculate_macd(data, short_window=12, long_window=26, signal_window=9):
    rows, cols = data.shape
    macd_values = np.empty((rows, cols))
    signal_line_values = np.empty((rows, cols))
    histogram_values = np.empty((rows, cols))

    for i in prange(cols):
        short_ema = np.zeros(rows)
        long_ema = np.zeros(rows)

        for j in range(1, rows):
            short_ema[j] = (data[j, i] - short_ema[j - 1]) * (2 / (short_window + 1)) + short_ema[j -
1]

            long_ema[j] = (data[j, i] - long_ema[j - 1]) * (2 / (long_window + 1)) + long_ema[j - 1]

        macd_values[:, i] = short_ema - long_ema

        signal_line = np.zeros(rows)
        for j in range(1, rows):
            signal_line[j] = (macd_values[j, i] - signal_line[j - 1]) * (2 / (signal_window + 1)) +
signal_line[j - 1]

        signal_line_values[:, i] = signal_line
        histogram_values[:, i] = macd_values[:, i] - signal_line

    return macd_values, signal_line_values, histogram_values
```

Bollinger Band

```
def calculate_bband(data, window=20, num_std_dev=2):
    num_rows, num_cols = data.shape
    upper_bands = np.zeros_like(data)
    lower_bands = np.zeros_like(data)
    mid_bands = np.zeros_like(data)

    for col in prange(num_cols):
        for i in prange(window - 1, num_rows):
            window_slice = data[i - window + 1 : i + 1, col]
            mid_bands[i, col] = np.mean(window_slice)
            std_dev = np.std(window_slice)
            upper_bands[i, col] = mid_bands[i, col] + num_std_dev *
std_dev     lower_bands[i, col] = mid_bands[i, col] - num_std_dev *
std_dev
    return upper_bands, mid_bands, lower_bands
```

$\mathrm{BOLU} = \mathrm{MA}(\mathrm{TP}, n) + m * \sigma[\mathrm{TP}, n]$

$\mathrm{BOLD} = \mathrm{MA}(\mathrm{TP}, n) - m * \sigma[\mathrm{TP}, n]$

where:

BOLU = Upper Bollinger Band

BOLD = Lower Bollinger Band

MA = Moving average

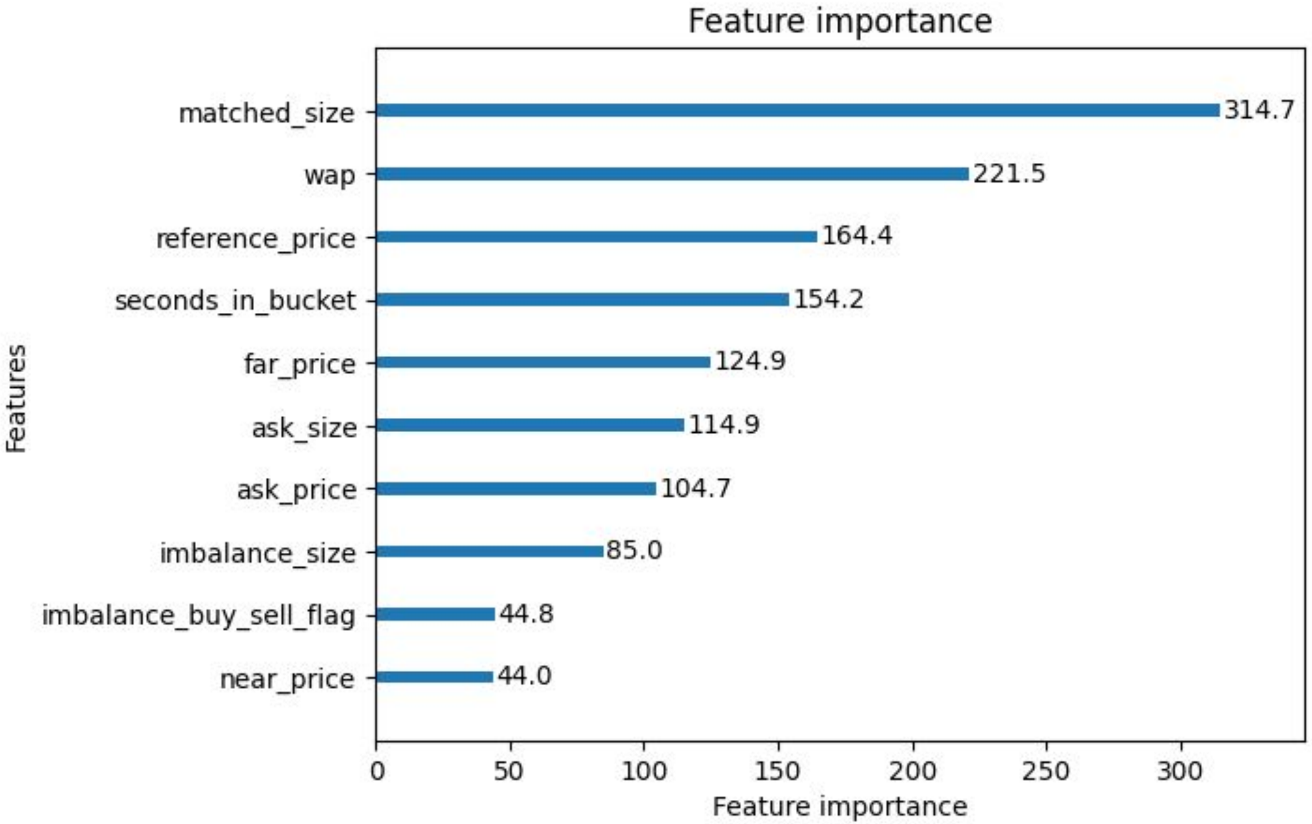
TP (typical price) = (High + Low + Close) ÷ 3

n = Number of days in smoothing period (typically 20)

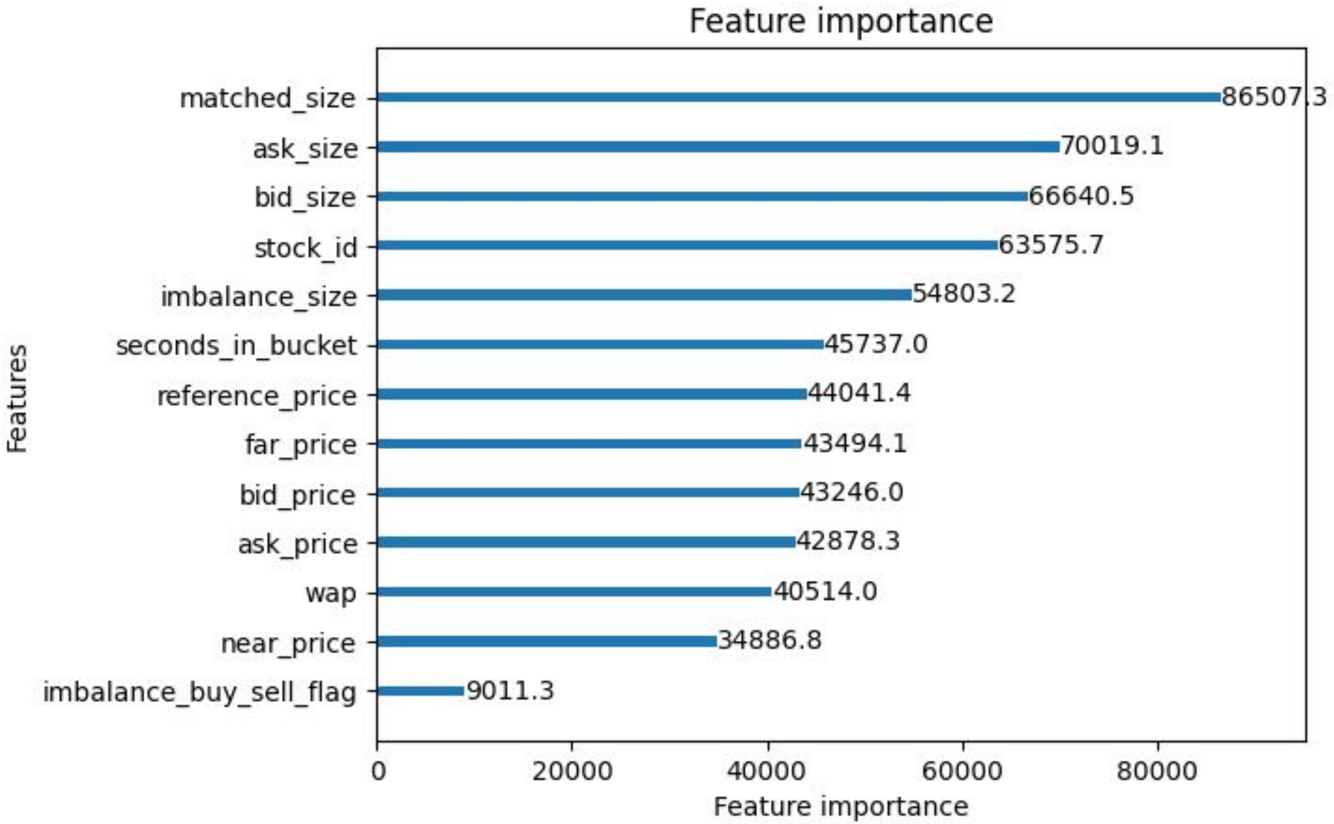
m = Number of standard deviations (typically 2)

$\sigma[\mathrm{TP}, n]$ = Standard Deviation over last n periods of TP

03. FEATURE IMPORTANCE



03.FEATURE IMPORTANCE



03. FEATURE IMPORTANCE